PURESTORAGE®
Uncomplicate Data Storage, Forever

WHITE PAPER

Toward a More Simple, Scalable HPC Storage Model

Democratizing HPC storage for all users.

Contents

Introduction

As long as there has been high-performance computing (HPC), there has been a need for high-performance storage. In the early days of supercomputers, most of their storage systems were built alongside the actual supercomputer itself, each one with a bespoke architecture to suit its particular needs. But over time, as the world of supercomputing grew, the need for more standardized approaches to storage became necessary. This began a move toward POSIX-compatible, ***clustered,*** or ***parallel*** file systems. But in the modern era of supercomputing, HPC, and machine learning, the complexity and fragility of this paradigm is becoming a huge burden on enterprises, research labs, and academia. Pure Storage believes there is a better approach to simple, scalable HPC storage.

What Drove the Need for Parallel File Systems?

Traditional Paradigms: NAS, SAN, and DAS

When analyzing, transforming, or generating huge amounts of data, any individual storage server or system can be overwhelmed by the massive needs of supercomputing. Whether this was traditional file-based network-attached storage (NAS), block-based storage area networks (SAN), or distributed direct-attached storage (DAS), all these models presented trade-offs and challenges for HPC workloads. While NAS provided extremely easy setup and data sharing, the nature of running on a single file server meant that any sizable HPC workload would hit a scalability bottleneck very quickly, as one could only build a single storage controller to be so large. SAN could present massive amounts of block storage to nodes, but the single-initiator nature of block storage meant that managing data sharing became a major challenge. Lastly, utilizing DAS offered ideal latency and scalability, but required constantly moving data between systems to where it is needed, and then only being able to utilize the storage performance of your local drives. Each of these paradigms proved unsuitable for the needs of HPC.

How Do Parallel File Systems Work?

Parallel file systems (PFSs) attempt to solve these problems by building large-scale storage pools using a highly distributed model that breaks up the different functions of a storage system into modular components. At its core, a file system does three things:

- Durably stores and retrieves data, generally in the form of **files**, and provides access through a standardized set of commands or APIs to **read, modify, rename, or write files**.
- Records metadata about those files, such as **hierarchical directory structures, permissions, filenames, and attributes**.
- Manages some kind of **database** to track where the data for each file is stored on whatever media it is managing.

In general, a PFS takes these various functions and breaks them up into components which can then be scaled independently across multiple nodes. Some PFSs may provide lightweight capabilities and rely on applications to manage certain functionality, while others may implement complex functions like distributed file locking capabilities.

What Are the Challenges of Parallel File Systems?

While a PFS can provide massive scale and performance, because of its highly distributed and configurable nature, it can require a tremendous amount of customization, tuning, and maintenance, all of which require specific expertise in the particular PFS that is being used. Some of the major challenges presented by a PFS are:

- Dedicated, complex, and specialized high-performance networking infrastructure is often also required to support the storage system's communication between these different nodes. These networks by themselves can represent a major cost of the overall system, especially if they are using technology that is not widely adopted outside of HPC.
- Some of these modular components must reside in the endpoint accessing the data, requiring components of your storage system to live within the operating system kernel of your compute nodes. This can lead to conflicts and complexity, as changes in your storage system software stack now can impact all your compute nodes. Requiring all components to change at once introduces tremendous disruption during upgrades, and image management of storage clients becomes critical to uptime and availability.
- Distribution and configuration of services, nodes, and data can have major impacts on performance of different workloads. While many PFSs offer incredible configurability, this complexity is—in general—left to the storage architects and administrators to handle. And while a PFS can be configured to offer massive throughput on large files, tuning it to also provide high IOPS on small files can sometimes require completely separate infrastructure or storage pools to accommodate these different workload profiles. Incorrect configuration or architecture can be difficult to change or replace non-disruptively, making the consequences of these errors very costly or painful.
- Because of this complexity, and because of the limited applicability of PFSs, the pool of expertise for installing, configuring, tuning, and managing these systems is limited. With the explosion of HPC environments brought about by both AI and enterprise HPC, these human resources are now in more demand than ever before.

However, for two decades, these trade-offs in complexity have been tolerated as a PFS was seen as the only viable solution to achieving the performance and scalability requirements of large HPC clusters.

Solving These Challenges in a New Way: Scale-out Network-attached Storage

Early Scale-out NAS Approaches

Outside of the HPC space, another approach to solving the scalability challenge faced by NAS systems was being explored by several different companies. By using the same access paradigm as a typical NAS system but transparently scaling out the system behind it, one could construct massive NAS volumes that offered high performance as well as very large namespaces.

Several different architectural approaches to accomplishing this were developed. Some leveraged traditional protocols such as NFS but utilized a round-robin approach to direct client connections to multiple targets, where each target presented a view of an identical namespace. Others relied on client-side protocol advances such as Parallel NFS (pNFS), first introduced in NFS version 4.1, to redirect portions of a namespace to different back-end storage systems. And some others added a layer of indirection in front of many different back-end systems, transparently integrating many individual namespaces into a larger virtual one.

However, systems implementing each of these approaches often suffered from deficiencies that were deemed unacceptable for certain HPC workloads. Some systems could handle massive throughput on many large files but struggled with lots of small files or metadata operations. Others might reach performance bottlenecks on a small portion of the namespace because the performance of that partition was limited to a single back-end storage server. Others required client-side services that relied on less performant NFS implementations. In each case, many HPC environments chose to remain on a parallel file system despite the added complexity required.

Solving the Scale-out Trade-offs: Pure Storage FlashBlade

In 2015, Pure Storage embarked on a project to solve the challenge of scale-out unstructured data storage. The disruptive power of NAND flash memory over its antediluvian predecessor the hard disk offered Pure Storage the opportunity to reimagine the architecture of a typical scale-out system. Taking inspiration from modern database architectures and some parallel file systems and leveraging native direct access to NAND flash, FlashBlade® was designed to:

- **Scale predictably** in both capacity and performance.
- Deliver that **performance across multiple dimensions**, including small- and large-file scenarios and high-throughput or high-IOPS workloads.
- Offer multiple access methods for unstructured data with **native file (NAS) and object storage (S3) protocol support**.
- **Protect** data integrity with state-of-the-art, automatic resiliency features.[1]
- Bring **simplicity** to customers for the first time in the scale-out storage market by using industry-standard protocols and integrated networking.
- Be **future-proof** with Pure Storage's unique commitment to **Evergreen® storage**.

For a deeper dive into the architecture of **Purity//FB**, FlashBlade's operating system, please refer to The Purity//FB Architecture (Pure1® login required) and "Unlocking the Power of Metadata Management with FlashBlade."

A Simpler Approach to HPC Storage

With the rapid pace of innovation in the HPC space ushered in by the rise of artificial intelligence and ever-increasing compute capability, a storage system that is complex, inflexible, and unreliable presents an unacceptable barrier to progress. Pure Storage® FlashBlade requires no invasive client-side software or provisioning, provides a single architecture that works both for HPC workloads as well as regular enterprise workloads, and does all this while still being simple to manage and consume. Both in academia and the enterprise, customers need a better approach to delivering scalable, high-performance storage solutions for today's HPC workloads. How does Pure Storage FlashBlade solve the storage challenges that plague both parallel file systems and traditional scale-out NAS architectures?

A Truly Scale-out, Modular Storage Architecture

Like a PFS, FlashBlade separates and scales the various logical functions of a storage system to avoid processing bottlenecks. One of the core philosophies of FlashBlade's design is to avoid any scale-up processes, or put another way: Every problem needs to have a scale-out solution. Otherwise, once your system grows to meet the ceiling of that single scale-up process, you've reached the apex of your system's scalability.

Purity//FB separates client access, metadata management, and data storage into scalable microservices, which are automatically scaled and distributed across all the blades in a FlashBlade system, offering the option of linear scalability of performance and capacity.

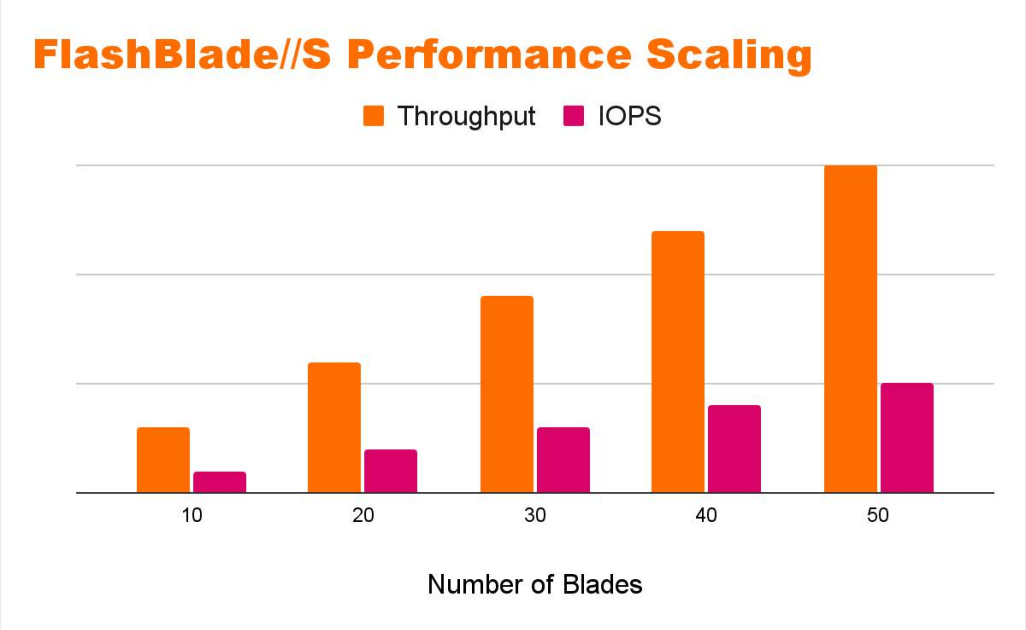

FIGURE 1 Performance increases linearly with the number of blades in a FlashBlade system.

Client Access: Protocol Endpoints

A FlashBlade system evenly distributes incoming client connections across a set of microservice instances called protocol endpoints, which service I/O requests for file or object storage. On FlashBlade//S™, these endpoints scale with the number of blades, offering additional front-end bandwidth and processing that scales linearly with the number of blades. The protocol endpoints maintain the client connection and, using a deterministic algorithm, ascertain which virtual partition(s) of the namespace the client request needs to access.

Similar to a parallel file system, this means that clients can connect to many blades simultaneously by leveraging multiple connections, without having to resort to localized POSIX clients. Users can consume storage using standard NAS protocols like NFSv3 or NFSv4.1 and still see scalability benefits as the number of clients and blades increases.

Metadata Management: Authorities

The namespace of a FlashBlade system is divided into logical partitions which are each managed by an **authority** microservice. Authorities are stateless processes that can run on any blade—there is no alignment between a blade and a particular authority. Authorities maintain internal key/value metadata structures that map client data addresses to the data's locations on flash. Each authority exclusively controls a segment of persistent storage media across every storage device in the system. Purity//FB leverages modern database approaches to track and manage metadata but is designed with domain-specific optimizations for a storage system based around NAND flash. For more information, please refer to "Better Science, Volume 2: Maps, Metadata, and the Pyramid."

Authorities are similar to the dedicated metadata servers of many parallel file systems; however, they're an integrated and automated part of the FlashBlade itself and therefore don't require user tuning or separate configuration. As the number of blades increases, each authority gets more performant as more compute resources can be dedicated to each one. Metadata performance is optimized, especially in the case of many small files because each authority can act independently on its partition of the namespace. And because the namespace is not partitioned by folder or share but by underlying physical addresses,[2] a single directory of millions of files can still leverage the metadata performance of the entire system. Similarly, very large files can span multiple authority partition boundaries and thus find similar advantages in relation to throughput.

Data Storage: Storage Managers

Every storage device in the FlashBlade spawns a microservice called a **storage manager**. This process handles all I/O to that particular device, services requests from authorities, and if necessary, delivers data to the protocol endpoint that originated the I/O request. Each authority owns one dedicated segment of storage within each storage manager. Therefore, the storage manager's only tasks are to service requests to read or write data and perform media management tasks such as garbage collection, wear leveling, and monitoring drive health.

Like the storage servers or "targets" in many parallel file systems, the storage manager is responsible for the actual reading and writing of data. But because they leverage Pure Storage's unique DirectFlash® technology, they eliminate many layers of indirection in the typical storage stack leveraging solid-state drives (SSDs), thus improving latency, throughput, and flash endurance. For an in-depth look into how DirectFlash works, please refer to "Better Science, Volume 1: Hardware and Software Co-design with DirectFlash."

Coming Full Circle: HPC Storage without Limits or Complexity

While the same challenges exist today in the HPC space as they did 20 years ago—but at previously unimaginable scale, there exists a superior solution for institutions and enterprises looking to deliver predictable, reliable, and scalable storage for modern HPC workloads. Pure Storage FlashBlade solves the same challenges as a parallel file system but without the burden of complexity. By taking a new approach to solving these challenges that incorporates the best of both the PFS and scale-out NAS architectures, FlashBlade offers customers a solution that avoids the disadvantages and trade-offs of either legacy option.

Learn More

- **Pure Storage HPC Solutions**
- **Pure Storage FlashBlade**

1 The Three Rs of Data Storage: Resiliency, Redundancy, and Rebuilds

2 Here, "physical address" refers to identifiers used by clients—{inode, offset} tuples for files, and ObjectID for objects.

PS2543-01-en 01/24